Combining Model-Driven Engineering and Cloud Computing

Hugo Bruneliere, Jordi Cabot, Frédéric Jouault

Combining Model-Driven Engineering and Cloud Computing

Hugo Brunelière, Jordi Cabot and Frédéric Jouault
AtlanMod, INRIA RBA Center & EMN
4 rue Alfred Kastler, 44307 Nantes, France
{hugo.bruneliere,jordi.cabot,frederic.jouault}@inria.fr

Abstract

Service-orientation and model-driven engineering are two of the most dominant software engineering paradigms nowadays. This position paper explores the synergies between them and show how they can benefit from each other. In particular, the paper introduces the notion of Modeling as a Service (MaaS) as a way to provide modeling and model-driven engineering services from the cloud.

1. Introduction

Model-driven engineering (MDE) is becoming the dominant software engineering paradigm to specify, develop and maintain software systems. In MDE, models are the primary artifact of the engineering process and are used, for instance, to (semi)automatically generate the implementation of the final software system.

At the same time, service-orientation is gaining popularity as the standard way of designing and deploying software applications over the internet, specially for distributed and e-commerce applications. With service-orientation, software is reorganized as a set of interacting services that client applications (e.g. other services) can execute on demand. This is also known as *Software as a Service* (SaaS).

Therefore, it is just natural that we wonder how both paradigms can be integrated and benefit from each other. In this paper we discuss two different collaboration scenarios between MDE and SaaS: 1 - using MDE for the development of SaaS applications (Section 2) and 2 - using SaaS to deploy modeling services in the cloud (Section 3).

2. MDE *for* the cloud

MDE techniques can facilitate and (semi)automate the development of new SaaS applications. Several approaches (see [5], [8], [1] and [6] as relevant examples) have already explored this possibility. Given the set of domain models (i.e. platform-independent models) of the system-to-be (possibly complemented with additional information regarding the system decomposition into a set of services by means of UML profiles or domain-specific languages) model-to-model and model-to-text transformations could be used to generate and deploy the software system as part of a service-oriented architecture (SOA). Unfortunately, there is not yet a consensus on the right set of models, languages, model transformations and software processes for the model-driven development of SaaS systems.

Morevoer, this just solves a small part of the problem. One of the main challenges SaaS faces to become mainstream is the problem of coping with all existing software, that should now be considered as legacy software (from a cloud point of view). This software needs to be evolved and adapted in order to be able to be executed as a service. This topic has not yet been studied but we believe model-driven reverse engineering techniques (such as our own MoDisco framework [7],[2]) can help in this matter. These techniques are able to discover and extract software models out of the artifacts of a legacy system. The idea would be to be able to produce the kind of software models needed to reimplement the system under the SaaS paradigm using the methods mentioned above.

3. MDE *in* the cloud: Modeling as a Service

We believe cloud computing can bring many benefits to MDE. In this sense, we would like to propose the *Modeling as a Service* (MaaS) initiative [1]. Similar to SaaS, MaaS would allow the deployment and on-demand execution of modeling and model-driven services over the Internet. We are confident that MaaS will contribute to widen even more the adoption of MDE among software practitioners. In what follows we comment some possible applications of MaaS:

- Creation of collaborative and distributed modeling

[1] This acronym has also been independently proposed in the context of the ENVISION project (http://www.envision-project.eu), with the meaning of *Models as a Service*, as a way to exchange environmental models

tools to allow the specification and sharing of software models among team members in real-time.

- Definition of modeling mash-ups as a combination of MDE services from different vendors.
- Availability of model transformation engines in the cloud to provide platform-independent model management services
- Improving scalability of MDE. Models of real-life applications (specially those obtained by reverse engineering of running systems) are usually very large. Modeling services in the cloud would ensure the scalability of MDE techniques in those scenarios.
- Facilitating Model execution and evolution. Moving code-generation and simulation services to the cloud would facilitate the deployment and evolution of software applications (regardless whether those applications are implemented as SaaS or not) and reduce their time to market. Designers could forget about setting up the infrastructure to compile and deploy the applications and would be able to rely on the cloud for that.
- Solving tool interoperability problems. Exchanging data (and metadata) among MDE tools is one of the major challenges of MDE nowadays. So far, the problem is being addressed by defining bridges among the tools (e.g. [3]) but MaaS would offer a more transparent and global solution to this problem. For instance, bridges could be defined as services automatically executed on demand by other services when incompatibility issues are detected.
- Distributed Global model management. Complex MDE projects involve several models (possibly conforming to different metamodels), model transformations, model injectors and projectors, ... MaaS would facilitate the manipulation of all these modeling artifacts in a distributed environment.

Research on all these topics is in very preliminary stages. The firsts web-based modeling tools (such as Creately [2] or Gliffy[3]) have just recently appeared. Some discussions about the benefits of using the cloud as a modeling repository (see the WebUML[4] initiative) or as a deployment platform (see the Enterprise Architect[5] blog) have also started but no concrete results have been provided and most services in our list have yet to be explored. As far as we know, our initiative would be the first systematic research project on the benefits and challenges of the MaaS approach.

[2] http://creately.com/
[3] http://www.gliffy.com/
[4] http://webuml.org/
[5] http://www.theenterprisearchitect.eu/

We are confident that the technical challenges to make MaaS a reality can be overcomed by learning from how SaaS has developed. Some parts of SOAs are directly applicable to modeling services but others should be adapted. In particular, we will need to agree on a description language for this kind of services (probably based on a predefined taxonomy/ontology) to facilitate the discovery and composition of services, on a common type systems for those services and to check whether current standards for web service orchestration and choreography can be reused in our context. We also need to develop parallel versions for the most time-consuming model manipulation algorithms (i.e. for transformation, verification,...) in order to benefit from the computing power of the cloud.

4. Conclusions

This paper has commented on the possible synergies between the model-driven engineering and service-orientation paradigms. Some research challenges that need to be solved in order to make this collaboration possible have also been mentioned. We plan to continue our work by further exploring and addressing these challenges.

References

[1] K. Baïna, B. Benatallah, F. Casati, and F. Toumani. Model-driven web service development. In A. Persson and J. Stirna, editors, *CAiSE*, volume 3084 of *LNCS*, pages 290–306. Springer, 2004.

[2] G. Barbier, H. Bruneliere, F. Jouault, Y. Lennon, and F. Madiot. MoDisco, a Model-Driven Platform to Support Real Legacy Modernization Use Cases. In *Information Systems Transformation: Architecture-Driven Modernization Case Studies*, pages 365–400. Morgan Kaufmann, 2010.

[3] H. Brunelire, J. Cabot, C. Clasen, F. Jouault, and J. Bzivin. Towards Model Driven Tool Interoperability: Bridging Eclipse and Microsoft Modeling Tools. In *6th European Conference on Modelling Foundations and Applications*, page to appear. LNCS, 2010.

[4] A. Dan and W. Lamersdorf, editors. *Service-Oriented Computing - ICSOC 2006, 4th International Conference, Chicago, IL, USA, December 4-7, 2006, Proceedings*, volume 4294 of *Lecture Notes in Computer Science*. Springer, 2006.

[5] S. K. Johnston and A. W. Brown. A model-driven development approach to creating service-oriented solutions. In Dan and Lamersdorf [4], pages 624–636.

[6] I. Manolescu, M. Brambilla, S. Ceri, S. Comai, and P. Fraternali. Model-driven design and deployment of service-enabled web applications. *ACM Trans. Internet Technol.*, 5(3):439–479, 2005.

[7] MoDisco. MoDisco Toolbox. `http://www.eclipse.org/gmt/modisco/toolBox/`.

[8] G. Ortiz and J. Hernández. Service-oriented model-driven development: Filling the extra-functional property gap. In Dan and Lamersdorf [4], pages 471–476.